In this issue

3  Analysing IFCID 6 records
13  Re-linking DB2 modules using SMP/E
17  Simulating a production environment – part 3
26  PLAN and PACKAGE management
48  DB2 news

© Xephon plc 1998
Contributions
Articles published in DB2 Update are paid for at the rate of £170 ($250) per 1000 words and £90 ($140) per 100 lines of code for original material. To find out more about contributing an article, without any obligation, please contact us at any of the addresses above and we will send you a copy of our Notes for Contributors.

DB2 Update on-line
Code from DB2 Update can be downloaded from our Web site at http://www.xephon.com; you will need the user-id shown on your address label.
Analysing IFCID 6 records

The buffer pool hit ratio cannot be 100%, because the initial load of data pages to the buffer takes some synchronous I/Os. My client’s shop has a daily production batch job that updates the DB2 database with a buffer pool hit ratio of over 90%, but the fact is that the 10% synchronous I/O actually accounts for 90% of the run-time of the job! The job will go down the tube, with much longer elapsed time, if there is contention on a DASD with other jobs. (The job shares a DASD pool with another application and dedicating volumes to this job and its own application may cause even worse contention with itself.) Average access time to a 3390 DASD, normally 15 msec, could easily double per I/O.

Since the run-time of the job is critical for the on-time start-up of the on-line application, the DBA has to come up with an idea to reduce and stabilize the elapsed time of the job.

There are different areas that can be looked at, but buffer pool tuning is a must and cannot be ignored. To find whether the current buffer pool hit ratio is optimal for the job, the DBA needs to know whether there are minimal or balanced re-read I/Os for the same pages of each tablespace or index.

IFCID 6 records show the read activity to the buffer pool in detail by DBID, OBID, and page number. By sorting the records by DBID, OBID, and page number, and analysing the re-read activity of pages that belong to the same tablespace or index, the DBA will be able to determine which object of the database needs more buffer space – even if the buffer pool size for the database as a whole looks enough.

Here are the steps to follow for analysing the ‘end read I/O trace’:

1. Run the DB2 performance trace for the duration of the batch job:

   ```
   START TRACE(PERFM) DEST(SMF) PLAN(the plan name) CLASS(30) IFCID(6)
   ```

   Stop the trace as soon as the job ends. The trace will create SMF record type 102 with the plan name of the job.

2. Extract the IFCID 6 records from the SMF dump dataset. In most
shops the SMF dump dataset is created daily, which means the IFCID 6 records may not be available until the following day.

JCL001 AND JCL002

JCL001, with a source code in SYSIN, will assemble and link-edit a program called SMFREAD. It extracts IFCID type 6 records from the SMF dataset by DBID, OBID, and page number. The DB2 subsystem-id and the SMF record type are hard-coded in the program.

Next, JCL002 will run the program SMFREAD with the local SMF dump dataset as input. The output is written to a new dataset. The second step of the job will sort the output data in ascending order of DBID, OBID, and page number.

JCL001

//JCL001 JOB (acct_info), 'pgmer_name'. NOTIFY=, <==== your job card info
// MSGCLASS=?,GROUP=,USER=
/*JOBPARM . . . .
//******************************************************************************
// ASSEMBLE AND LINKEDIT
//******************************************************************************
//ASMH EXEC ASMHCL
//C.SYSLIB DD DSN=SYS1.MACLIB,DISP=SHR
// DD DSN=hlq.DB2.SDSNMACS,DISP=SHR <==== your high-level qualifier
//L.SYSIN DD *
SMFREAD CSECT
******************************************************************************
* THIS PROGRAM EXTRACTS IFCID 6 RECORD FROM THE DAILY SMF DUMP.       *
* SMF record type, IFCID ID, DB2 subsystem ID are hard-coded.           *
******************************************************************************
R1 EQU 1
R2 EQU 2
R3 EQU 3
R4 EQU 4
R5 EQU 5
R6 EQU 6
R7 EQU 7
R8 EQU 8
R9 EQU 9
R10 EQU 10
R11 EQU 11
R12 EQU 12
R13 EQU 13
R14 EQU 14
R15 EQU 15
*
BEGIN    SAVE (14,12)
BALR  R3,Ø
USING *,R3
USING SM1Ø2,R11
USING QWØØØ6,R5 DATA SECTION
ST  R13,SAVEAREA+4
LA  R13,SAVEAREA
OPEN (SMFREC,INPUT,IFCID6,OUTPUT)
GETMAIN RU,LV=32722
LR  R11,R1
READREC GET SMFREC,Ø(R11)
CLI SM1Ø2RTY,X'66' RECORD TYPE OF PERF, AUDIT OR MONITOR ?
BNE READREC
CLC SM1Ø2SSI,=C'DBxx' DB2 ssid
BNE READREC
L  R4,SM1Ø2END
LA  R1Ø,Ø(R11,R4)
CLC 4(2,R1Ø),=X'0006' IFCID 6 ?
BNE READREC
LA  R5,SM1Ø2END+16
MVC DBID,QWØØØ6DB
MVC OBID,QWØØØ6OB
MVC BPID,QWØØØ6BP
MVC PAGEID,QWØØØ6PN
MVC READTYPE,QWØØØ6F
MVC ROSTOR,QWØØØ6AC
PUT IFCID6,WORKAREA
B READREC
EOFILE CLOSE (SMFREC,,IFCID6)
L  R13,SAVEAREA+4
RETURN (14,12)
SMFREC DCB DSORG=PS,RECFM=VBS,MACRF=GM,BLKSIZE=27998, X
LRECL=3276Ø,DDNAME=SMFIN,EODAD=EOFILE
IFCID6 DCB DSORG=PS,RECFM=FB,MACRF=PM,DDNAME=OUTREC,LRECL=8Ø, X
BLKSIZE=312Ø
*
DS  ØF
SAVEAREA DS  18F
WORKAREA DS  ØCL8Ø
DBID DS  XL2 DBID
OBID DS  XL2 PAGESET OBID
BPID DS  F BUFFER POOL INTERNAL ID
PAGEID DS  XL3 1ST PAGE NUMBER TO BE READ
READTYPE DS  C
ROSTOR DS  F
DC  64C' '
DS  ØF

TO RUN A REPORT JOB

The following REXX program will take the output dataset from step 2 as an input argument. It processes the IFCID 6 records to create a report that shows the buffer REREAD RATIO for each tablespace and index.
This program reports the following information using the input data from ifcid 6 records.

- dbid, obid, buffer pool
- total number of reads, number of pages read once
- number of reads of same pages
- number of pages read multiple times
- number of rereads, number of reads per page
- reread ratio

As summary,
- total number of read I/O, rereads
- number of pages per page, reread ratio

ARG indsn
address tso
"alloc ddname(ifcid6) dsname('"indsn"') shr reuse"
"EXECIO * diskr ifcid6 (finis"
NUMERIC DIGITS 7
if queued() <> Ø then do
  say ' ' ' ' ' ' ' TOTAL # ' '# PAGES ','
  --- REREAD --- ' ' ' ' ' REREAD'
  say 'DBID ' 'OBID ' 'BP ' ' OF READ ' 'READ ONCE ' ,
  #READ #PAGES ' ' #REREAD ' 'READ/PG ' 'RATIO '
  say '— ' '— ' '— ' '‘ ‘ ’ ‘ ‘ ‘ ‘ ‘ ,
  ‘ ‘ ‘ ‘ ‘ ‘ ‘ ‘ ,
end
else EXIT

k=1 /* used for index of the bp */
/* initialize output variable */
dbid. = '
obid. = '
bpid. = '' /* contains the buffer pool number */
pageid. = ''
readtyp. = ''
reread. = Ø /* total 1 reads of an obid */
nr1read = Ø /* total non-one reads of an obid */
totread = Ø /* total reads of the obid */
totrread = Ø /* total rereads of the obid */
nn1pages = Ø /* total pages that had >1 reads */
bptread. = Ø /* total reads by bp pool and the grand total */
bptrread.= Ø /* total rereads by bp pool and the grand total */
/* process the first record to init the parms */
PARSE PULL record1
do while SUBSTR(record1,12,1) <> 'R'
    PARSE PULL record1
end

curdbid = SUBSTR(record1,1,2)
curobid = SUBSTR(record1,3,2)
curbpid = SUBSTR(record1,5,4)
curpgid = SUBSTR(record1,9,3)
reread = 1
bpid.1 = curbpid

do until queued() = Ø

    PARSE PULL record1
    if SUBSTR(record1,12,1) = 'R' then do
        hexdbid = SUBSTR(record1,1,2)
        if hexdbid = curdbid then do
            hexobid = SUBSTR(record1,3,2)
            if hexobid = curobid then do
                hexpgid = SUBSTR(record1,9,3)
                if hexpgid = curpgid then
                    reread = reread + 1
                else do
                    if reread <> 1 then do
                        nn1read = nn1read + reread
                        nn1pages = nn1pages + 1
                    end
                    else nn1read = nn1read + 1
                    reread = 1
                    curpgid = hexpgid
                end
                else do
                    call sayrread
                    curobid = hexobid
                    curbpid = SUBSTR(record1,5,4)
                    curpgid = SUBSTR(record1,9,3)
                    reread = 1
                end
            else do
                call sayrread
                curdbid = hexdbid
                hexobid = SUBSTR(record1,3,2)
                curobid = hexobid
                curbpid = SUBSTR(record1,5,4)
                hexpgid = SUBSTR(record1,9,3)
                curpgid = hexpgid
            end
        else do
            call sayrread
            curdbid = hexdbid
            hexobid = SUBSTR(record1,3,2)
            curobid = hexobid
            curbpid = SUBSTR(record1,5,4)
            hexpgid = SUBSTR(record1,9,3)
            curpgid = hexpgid
        end
    end
end
else do
    call sayrread
    curdbid = hexdbid
    hexobid = SUBSTR(record1,3,2)
    curobid = hexobid
    curbpid = SUBSTR(record1,5,4)
    hexpgid = SUBSTR(record1,9,3)
    curpgid = hexpgid
reread = 1
end
end
end

/* write an output line */
call sayrread

/* write the summary line */
say ''
say ''
say ''
say ''
say ''
say ''
say ' BP ' 'TOTAL READS ' 'TOTAL REREADS ' 'READ/PG ' 'RATIO'
say ' — ' '————— ' '—————— ' '——— ' '——'
grtread = Ø
grtrread = Ø
do i = 1 to k
  bpidnum = RIGHT(C2D(bpid.i),2)
  bptrnum = RIGHT(bptread.i,10)
  bptrrnm = RIGHT(bptrread.i,9)
  bprppg = (bptread.i + bptrread.i) / (bptread.i - bptrread.i)
  bprppg = FORMAT(bprppg,6,1)
  bpratio = bptrread.i / bptread.i
  bpratio = FORMAT(bpratio,3,2)
  say ' ' bpidnum ' ' bptrnum ' ' bptrrnm bprppg bpratio
  grtread = grtread + bptread.i
  grtrread = grtrread + bptrread.i
end
grtrnum = RIGHT(grtread,11)
grpprm = RIGHT(grtrread,10)
grpp = (grtread + grtrread) / (grtread - grtrread)
grpp = FORMAT(grpp,6,1)
gratio = grtrread / grtread
gratio = FORMAT(gratio,3,2)
say ' TOTAL ' grtrnum ' ' grpprm grpp gratio

address tso
"FREE fi(ifcid6)"

EXIT

sayrread :
  if reread <> 1 then do
    nnlread = nnlread + reread
    nnlpages = nnlpages + 1
  end
  else nlread = nlread + 1
dbidnum = RIGHT(C2D(curdbid),4)
obidnum = RIGHT(C2D(curobid),4)
bpidnum = RIGHT(C2D(curbpid),2)
totread = n1read + nn1read
totrread = nn1read - nnlpages
totrnum = RIGHT(totread,8)
totrrnum = RIGHT(totrread,7)
n1rnum = RIGHT(n1read,8)
nn1rnum = RIGHT(nn1read,8)
nnlpnum = RIGHT(nnlpages,6)
readppg = (n1read + nn1read) / (nnlpages + n1read)
rdppgnum = FORMAT(readppg,6,Ø)
ratio = totrread / totread
rationum = FORMAT(ratio,1,2)
say dbidnum ' ' obidnum ' ' bpidnum ' ' totrnum ' ' n1rnum ' ',
nn1rnum ' ' nnlpnum ' ' totrrnum ' ' rdppgnum ' ' rationum

/* accumulate the read stats for each bp */
do i = 1 to k
  if bpid.i = curbpid then do
    bptread.i = bptread.i + totread
    bptrread.i = bptrread.i + totrread
    leave
  end
end

if i > k then do /* new bpid */
  k = k + 1
  bpid.k = curbpid
  bptread.k = totread
  bptrread.k = totrread
end

n1read = Ø
nnlpages = Ø
nnlread = Ø
RETURN

ANALYSING THE RE-READ REPORT

Figure 1 shows the actual report indicating all the re-read information
of the batch job. Figure 2 shows the summary by buffer pool.

The headings for the report shown in Figure 1 are as follows:

- **BP** – the buffer pool-id assigned to the DBID and OBID.
- **Total # of read** – the total number of read I/Os to the buffer for the
  DBID and OBID.
<table>
<thead>
<tr>
<th>DBID</th>
<th>OBID</th>
<th>BP</th>
<th>TOTAL # OF READ</th>
<th># PAGES READ ONCE</th>
<th>— REREAD — #READ</th>
<th>#READ/RATIO</th>
<th>REREAD #PAGES</th>
<th>REREAD/RATIO</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>95</td>
<td>0</td>
<td>74</td>
<td>68</td>
<td>6</td>
<td>3</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>100</td>
<td>0</td>
<td>20</td>
<td>0</td>
<td>20</td>
<td>3</td>
<td>17</td>
<td>7</td>
</tr>
<tr>
<td>6</td>
<td>9</td>
<td>0</td>
<td>38</td>
<td>38</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>10</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>96</td>
<td>0</td>
<td>3</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>101</td>
<td>0</td>
<td>13</td>
<td>13</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>7</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>7</td>
<td>6</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>28Ø</td>
<td>2</td>
<td>1</td>
<td>20078</td>
<td>15947</td>
<td>4131</td>
<td>1952</td>
<td>2179</td>
<td>1</td>
</tr>
<tr>
<td>28Ø</td>
<td>4</td>
<td>1</td>
<td>21639</td>
<td>16293</td>
<td>5346</td>
<td>2439</td>
<td>2907</td>
<td>1</td>
</tr>
<tr>
<td>28Ø</td>
<td>6</td>
<td>1</td>
<td>1877</td>
<td>1633</td>
<td>244</td>
<td>120</td>
<td>124</td>
<td>1</td>
</tr>
<tr>
<td>28Ø</td>
<td>10</td>
<td>1</td>
<td>34</td>
<td>19</td>
<td>15</td>
<td>7</td>
<td>8</td>
<td>1</td>
</tr>
<tr>
<td>28Ø</td>
<td>12</td>
<td>1</td>
<td>4365</td>
<td>3511</td>
<td>854</td>
<td>410</td>
<td>444</td>
<td>1</td>
</tr>
<tr>
<td>28Ø</td>
<td>14</td>
<td>2</td>
<td>473</td>
<td>469</td>
<td>4</td>
<td>2</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>28Ø</td>
<td>18</td>
<td>2</td>
<td>13459</td>
<td>5151</td>
<td>8308</td>
<td>3102</td>
<td>5206</td>
<td>2</td>
</tr>
<tr>
<td>28Ø</td>
<td>24</td>
<td>1</td>
<td>241</td>
<td>95</td>
<td>146</td>
<td>60</td>
<td>86</td>
<td>2</td>
</tr>
<tr>
<td>28Ø</td>
<td>26</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>28Ø</td>
<td>28</td>
<td>1</td>
<td>5</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>28Ø</td>
<td>30</td>
<td>2</td>
<td>4122</td>
<td>3144</td>
<td>978</td>
<td>446</td>
<td>532</td>
<td>1</td>
</tr>
<tr>
<td>28Ø</td>
<td>38</td>
<td>1</td>
<td>4185</td>
<td>3546</td>
<td>559</td>
<td>276</td>
<td>283</td>
<td>1</td>
</tr>
<tr>
<td>28Ø</td>
<td>42</td>
<td>1</td>
<td>37</td>
<td>11</td>
<td>26</td>
<td>10</td>
<td>16</td>
<td>2</td>
</tr>
<tr>
<td>28Ø</td>
<td>44</td>
<td>1</td>
<td>39</td>
<td>27</td>
<td>12</td>
<td>5</td>
<td>7</td>
<td>1</td>
</tr>
<tr>
<td>28Ø</td>
<td>50</td>
<td>2</td>
<td>10</td>
<td>10</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>28Ø</td>
<td>52</td>
<td>2</td>
<td>43</td>
<td>11</td>
<td>32</td>
<td>10</td>
<td>22</td>
<td>2</td>
</tr>
<tr>
<td>28Ø</td>
<td>54</td>
<td>2</td>
<td>13</td>
<td>7</td>
<td>6</td>
<td>2</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>28Ø</td>
<td>56</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>28Ø</td>
<td>78</td>
<td>2</td>
<td>17</td>
<td>17</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>28Ø</td>
<td>80</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>28Ø</td>
<td>122</td>
<td>3</td>
<td>19740</td>
<td>6724</td>
<td>13016</td>
<td>5494</td>
<td>7522</td>
<td>2</td>
</tr>
<tr>
<td>28Ø</td>
<td>124</td>
<td>3</td>
<td>40</td>
<td>10</td>
<td>30</td>
<td>10</td>
<td>20</td>
<td>2</td>
</tr>
<tr>
<td>28Ø</td>
<td>126</td>
<td>3</td>
<td>2262</td>
<td>1385</td>
<td>877</td>
<td>394</td>
<td>483</td>
<td>1</td>
</tr>
<tr>
<td>28Ø</td>
<td>128</td>
<td>3</td>
<td>12</td>
<td>0</td>
<td>12</td>
<td>5</td>
<td>7</td>
<td>2</td>
</tr>
<tr>
<td>28Ø</td>
<td>132</td>
<td>3</td>
<td>33</td>
<td>8</td>
<td>25</td>
<td>6</td>
<td>19</td>
<td>2</td>
</tr>
<tr>
<td>28Ø</td>
<td>134</td>
<td>3</td>
<td>6</td>
<td>6</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>28Ø</td>
<td>136</td>
<td>3</td>
<td>0</td>
<td>15</td>
<td>16</td>
<td>5</td>
<td>11</td>
<td>2</td>
</tr>
<tr>
<td>28Ø</td>
<td>138</td>
<td>3</td>
<td>4075</td>
<td>2044</td>
<td>2031</td>
<td>914</td>
<td>1117</td>
<td>1</td>
</tr>
<tr>
<td>28Ø</td>
<td>142</td>
<td>3</td>
<td>4656</td>
<td>1969</td>
<td>2687</td>
<td>1165</td>
<td>1522</td>
<td>1</td>
</tr>
<tr>
<td>28Ø</td>
<td>148</td>
<td>3</td>
<td>16028</td>
<td>381</td>
<td>16447</td>
<td>4338</td>
<td>12109</td>
<td>4</td>
</tr>
<tr>
<td>28Ø</td>
<td>150</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>28Ø</td>
<td>152</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>28Ø</td>
<td>156</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>28Ø</td>
<td>164</td>
<td>3</td>
<td>19</td>
<td>10</td>
<td>9</td>
<td>2</td>
<td>7</td>
<td>2</td>
</tr>
<tr>
<td>28Ø</td>
<td>166</td>
<td>3</td>
<td>71</td>
<td>67</td>
<td>4</td>
<td>2</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>28Ø</td>
<td>174</td>
<td>3</td>
<td>7</td>
<td>7</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>28Ø</td>
<td>178</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>28Ø</td>
<td>180</td>
<td>3</td>
<td>464</td>
<td>378</td>
<td>86</td>
<td>35</td>
<td>51</td>
<td>1</td>
</tr>
<tr>
<td>28Ø</td>
<td>184</td>
<td>3</td>
<td>4290</td>
<td>2900</td>
<td>1390</td>
<td>654</td>
<td>736</td>
<td>1</td>
</tr>
<tr>
<td>28Ø</td>
<td>190</td>
<td>3</td>
<td>28187</td>
<td>11844</td>
<td>8263</td>
<td>3779</td>
<td>4484</td>
<td>1</td>
</tr>
<tr>
<td>28Ø</td>
<td>192</td>
<td>3</td>
<td>44</td>
<td>6</td>
<td>38</td>
<td>17</td>
<td>21</td>
<td>2</td>
</tr>
<tr>
<td>28Ø</td>
<td>194</td>
<td>3</td>
<td>21</td>
<td>5</td>
<td>16</td>
<td>8</td>
<td>8</td>
<td>2</td>
</tr>
<tr>
<td>28Ø</td>
<td>198</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>28Ø</td>
<td>202</td>
<td>3</td>
<td>344</td>
<td>342</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>28Ø</td>
<td>206</td>
<td>3</td>
<td>3586</td>
<td>859</td>
<td>2727</td>
<td>975</td>
<td>1752</td>
<td>2</td>
</tr>
<tr>
<td>28Ø</td>
<td>208</td>
<td>3</td>
<td>9</td>
<td>9</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>28Ø</td>
<td>212</td>
<td>3</td>
<td>8</td>
<td>8</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>28Ø</td>
<td>214</td>
<td>3</td>
<td>2888</td>
<td>1011</td>
<td>1789</td>
<td>696</td>
<td>1093</td>
<td>2</td>
</tr>
<tr>
<td>28Ø</td>
<td>216</td>
<td>3</td>
<td>35</td>
<td>13</td>
<td>22</td>
<td>8</td>
<td>14</td>
<td>2</td>
</tr>
<tr>
<td>28Ø</td>
<td>218</td>
<td>3</td>
<td>93</td>
<td>93</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>28Ø</td>
<td>220</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

**Figure 1: Buffer pool re-read ratio report**
• # Pages read once – the number of pages of the DBID and OBID that didn’t get read to the buffer more than once. After the initial load the pages were read from the buffer when the program requested GETPAGES.

• Reread #read – the number of read I/Os for a page of the DBID and OBID.

• Reread #pages – the number of pages that were read to the buffer more than once.

• #reread – the number of read I/Os for a page that had been read to the buffer.

• Read/page – the average number of read I/Os per page (total # of read) / ( # pages read once + reread #pages).

• Reread ratio – #reread / total # of read.

While the buffer pool hit ratio of BP1, BP2, and BP3 are all over 90% for this job according to the DB2PM report, this report discloses a possible problem with the performance of BP3.

If several runs of this job keep showing high values for the ‘reread ratio’ of OBID 148 and 206 for DBID 280, it could be a good time to put more effort into the buffer pool tuning.

Samuel Park  
DBA (USA)  
© Xephon 1998
Re-linking DB2 modules using SMP/E

With the delivery of DB2 Version 4.1, there may be instances where the re-linking of DB2 supplied modules is necessary. The first instance is prompted by IBM DB2 development, as specified in the DATABASE 2 for MVS/ESA Version 4 Installation Guide (Document Number SC26-3456-00):

2.7.1.11 DSNALI Linkage Attributes

DSNALI is now shipped with the attributes AMODE(31) and RMODE(ANY). You must re-link DSNALI if your applications need to load it below the 16MB line.

We encountered the second scenario when we migrated our production DB2 Version 3.1 subsystem to Version 4.1 and received the following error message when DSNECP10 was called by a COBOL applications program:

+ IKF995I ILBONTR AMODE ERROR. ENTERED IN AMODE 31.

In the case of my current shop, both DSNALI and DSNECP10 had to be re-linked with AMODE(24) and RMODE(24). There are two ways of doing this, either manually or using SMP/E.

Whether you are using the manual method or the SMP/E method, the first step is the same – an SMP/E unload of the LMOD entry that we wish to re-link. In this case, when we issued the command ‘UNLOAD LMOD(DSNALI,DSNECP10)’ in our DB2 target zone we received:

```
REP       LMOD            ( DSNALI   )
            LASTUPD         ( HDB441Ø )
            LASTUPDTYPE     ( ADD )
            SYSLIB          ( SDSNLOAD )
            /* LEPARM    */   STD
            ++LMODIN
            ORDER DSNAA
            ENTRY DSNALI
            ALIAS DSNHLI2
            ALIAS DSNWLI2
            MODE AMODE(24),RMODE(24)
            ++ENDLMODIN

REP       LMOD            ( DSNECP1Ø )
```

Using the linkage editor statements supplied in the ++LMODIN section, it was very easy to re-link the two modules, and with that the manual re-link process is complete.

The problem here is that, if we ever apply maintenance to any object module referenced by the two modules, they are re-linked automatically by SMP/E with AMODE(31) RMODE(ANY) – the linkage editor attributes listed in SMP/E for those two load modules. To ensure that this modification is carried across maintenance levels, we must make SMP/E aware of the changes that we’ve made. To do that, we use SMP/E usermods.

In addition to the LMOD information that we’ve acquired above, we also issue the following command in our DB2 target zone:

```
UNLOAD MOD(DSNALI, DSNECP10)
```

From which we receive the following output:

```
UCLIN .
REP  MOD      ( DSNALI )
     LASTUPD   ( HDB4410 )
     LASTUPDTYPE ( ADD )
     DISTLIB   ( ADSNLOAD )
     FMID      ( HDB4410 )
     RMID      ( CSHMOD2 )
     LMOD      ( DSNALI  DSNX9SER  DSNX9STP )

REP  MOD      ( DSNECP10 )
     LASTUPD   ( HDB4410 )
     LASTUPDTYPE ( ADD )
     DISTLIB   ( ADSNLOAD )
     FMID      ( HDB4410 )
     RMID      ( CSHMOD1 )
     LMOD      ( DSNECP10 )
```
C.SHMOD1
The information from the MOD and LMOD entries is used to build our USERMODs. For DSNALI, we created C.SHMOD1:

++USERMOD(C.SHMOD2) REWORK(1998001) /* 24/24 INPLACE OF 31/ANY */ .
++VER(P115) FMID(HDB4410) .
++JCLIN .
//LKED EXEC PGM=IEWL
//SYSLMOD DD DISP=SHR,DSN=DSN410.SDSNLOAD
//SYSIN DD *
ORDER DSNAA
ENTRY DSNALI
ALIAS DSNHLI2
ALIAS DSNWLI2
MODE AMODE(24),RMODE(24)
NAME DSNALI(R)
++MOD(DSNALI)
   DISTLIB ( ADSNLOAD )
   LMOD ( DSNALI DSNX9SER DSNX9STP )
   TXLIB ( ADSNLOAD )

C.SHMOD2
For DSNECP2, we created C.SHMOD2:

++USERMOD(C.SHMOD1) REWORK(1998001) /* 24/24 IN PLACE OF 31/ANY */ .
++VER(P115) FMID(HDB4410) PRE(UN90165) .
++JCLIN .
//LKED EXEC PGM=IEWL
//SYSLMOD DD DISP=SHR,DSN=DSN410.SDSNLOAD
//SYSIN DD *
ORDER DSNAA /* added for KFF0365. RDD */
ENTRY DSNECP10
MODE AMODE(24),RMODE(24)
NAME DSNECP10(R)
++MOD(DSNECP10)
   DISTLIB ( ADSNLOAD )
   LMOD ( DSNECP10 )
   TXLIB ( ADSNLOAD )

The ++JCLIN information comes directly from the ++LMODIN section of the LMOD entries. As for the ++MOD information, the
FMID and PRE come from the MOD entries, as does the DISTLIB, LMOD, and TXTLIB information.

Once the SMP/E usermods are built, they can be received into the GLOBAL zone, where they can be applied to the DB2 target zones. You will notice that the linkage editor attribute changes to AMODE(24) and RMODE(24) when they are applied – but only for the LMODs specified in the USERMODs. Although modules DSNX9SER and DSNX9STP are specified in the LMOD list of object module DSNALI, they are still linked with AMODE(31) and RMODE(ANY).

When the time comes to receive and apply maintenance, we have to use SMP/E RESTORE to back out these two USERMODs before any new maintenance can be applied to the modules DSNALI and DSNECP10.

After we have applied our maintenance, we have to retrofit the USERMODs and let SMP/E know that the new maintenance level is recognized. We accomplish this by unloading the MOD entries for DSNALI and DSNECP10 once again, and then applying the LASTUPD information into the PRE information of the USERMODs.

One item of note – the first time maintenance is applied, the new object modules for DSNALI and DSNECP10 must be put into an object module library known to SMP/E, and the TXLIB operand changed, then the USERMODs re-received. This is because we are currently loading the object code for DSNALI and DSNECP10 out of the ADSNLOAD library, since all the maintenance has been accepted.

After we apply maintenance, the modules in ADSNLOAD will become back level until the new maintenance is accepted.

Chorng S (Jack) Hwang  
Principal  
HSA Systems (USA)  

© Xephon 1998

*DB2 Update* is looking for REXX EXECs, macros, program code, etc, that experienced DB2 users have written to make their life easier. Articles can be sent or e-mailed to Robert Burgess at any of the addresses shown on page 2.
Simulating a production environment – part 3

This month we conclude the article on simulating a production environment in DB2.

/*******  DISPLAYS SPECIFIC FILE RECORD *********/

LIST.SCR=LEFT(TBNAME,8) ’ ’LEFT(COLNAME,18) ’ ’LEFT(COLVALUE,40),
 ’ ’RIGHT(FREQUENC,4,0)
SAY ’TABLE COLUMN COLVALUE FREQUENCE’
SAY LIST.SCR
PARSE PULL .+9 .+19 COLVALUE +41 FREQUENC +5 .

/******* PREPARES CHANGED FIELDS TO REWRITE FILE RECORD ******/

IF FREQUENC¬=''
  THEN
    DO
      TBNAME=LEFT(TBNAME,18)
      COLNAME=LEFT(COLNAME,18)
      COLVALUE=LEFT(COLVALUE,254)
      IF FREQUENC¬='FFFF' &,
         DATATYPE(FREQUENC)=NUM THEN
        FREQUENC=D2C(FREQUENC,2)
      ELSE
        IF FREQUENC='FFFF' THEN
          FREQUENC=COPIES(X2C('FF'),2)
        ELSE
          DO
            A=A-1
            SAY ’***ERROR: FREQUENC IS INVALID’
            ITERATE
          END
    END

/******* PROCESSES NEXT RECORD ON INPUT FILE ******/

DO
  NM=NM+1
  TBLO.NM=TRANSLATE(TBNAME),
           ||TRANSLATE(COLNAME),
           ||TRANSLATE(COLVALUE),
           ||LEFT(FREQUENC,4,0)
END
END
ELSE
  NOP
IF VARI='YES' THEN
  NOP  

ELSE
   DO
      A=A+1
      PARSE VALUE TBL.A WITH 1 TBNAME 7.
      IF VAR2 = TBNAME THEN
         DO
            A=A-1
            ITERATE
         END
      ELSE
         DO
            SAY 'WHAT TABLE DO YOU WANT TO CHANGE?'
            PULL VAR2
            A=1
            ITERATE
         END
      END
   END

"ALLOC F("DD2") SHR REUSE DA("DSN2")"
"EXECIO "NM" DISKW "DD2" (FINIS STEM TBLO.)"
"FREE F("DD2")"
RETURN
/* REXX */

DB2CATCL
DB2CATCL displays values from the SYSCOLUMNS output file produced by the catalog extraction program PCATV3EX.
/* REXX */

DD1="I"TIME("S")
DD2="O"TIME("S")
DSN1="'XXXXXXXX.SYSCOLMS'
DSN2="'XXXXXXXX.SYSCOLOS'

"ALLOC F("DD1") SHR REUSE DA("DSN1")"
"EXECIO * DISKR "DD1" (FINIS STEM TBL.)"
"FREE F("DD1")"

/***** READS THRU INPUT FILE UNTIL SPECIFIED RECORD IS REACHED *****/
NM=Ø
SAY 'DO YOU WANT THE WHOLE DATABASE?'
PULL VAR1
IF VAR1='YES' THEN
   NOP
ELSE
DO
SAY 'WHAT TABLE DO YOU WANT TO CHANGE?'
PULL VAR2
END
DO A=1 TO TBL.Ø
  PARSE VALUE TBL.A WITH 1  TBNAME 7 ..
    19 NAME 37 ..
    37 COLCARD  41 ..
    46 LOW2KEY  54 ..
    54 HIGH2KEY 62 ..
    62 COLTYPE  70 .
  DO
    IF VAR1='YES' THEN
      NOP
    ELSE
      IF VAR2 = TBNAME THEN
        NOP
      ELSE
        ITERATE
    END
END

/*******  PROCESSES COLCARD VALUES ******/
DO
  COLCARD=C2X(COLCARD)
  IF COLCARD='FFFFFFFF' THEN
    DO
      A=A+1
      PARSE VALUE TBL.A WITH 1  TBNAME 7 .
      IF VAR2 = TBNAME THEN
        DO
          A=A-1
          ITERATE
        END
      ELSE
        DO
          SAY 'WHAT TABLE DO YOU WANT TO CHANGE?'
          PULL VAR2
          A=1
          ITERATE
        END
    END
  ELSE
    COLCARD=X2D(COLCARD,8)
  END

/*******  PROCESSES ALL TYPES OF LOW2KEY VALUES ******/
DO
  IF COLTYPE='DATE' THEN
    DO
      LOW2KEY=LEFT(C2X(LOW2KEY),8)
    END
LOW2KEY=INSERT(LOW2KEY,'        ')

ELSE
  DO
    IF COLTYPE='TIMESTAMP' THEN
      LOW2KEY=LEFT(C2X(LOW2KEY),16)
    ELSE
      DO
        IF COLTYPE='TIME' THEN
          DO
            LOW2KEY=LEFT(C2X(LOW2KEY),6)
            LOW2KEY=INSERT(LOW2KEY,'        ')
          END
        ELSE
          IF COLTYPE = 'CHAR'|,'VARCHAR' THEN
            DO
              LOW2KEY=LEFT(TRANSLATE(LOW2KEY,”:”,’ØØ’x),8)
              LOW2KEY=INSERT(LOW2KEY,'        ')
            END
          ELSE
            IF COLTYPE = 'DECIMAL' THEN
              DO
                LOW2KEY=LEFT(SPACE(LOW2KEY,1,’x’),17,’x’)
                LOW2KEY=STRIP(LOW2KEY,T,’x’)
                LOW2KEY=LEFT(C2X(LOW2KEY),17)
                LOW2KEY=STRIP(LOW2KEY,L,’F’)
              END
            ELSE
              IF COLTYPE = 'INTEGER' THEN
                DO
                  LOW2KEY=LEFT(SPACE(LOW2KEY,1,’x’),8,’x’)
                  LOW2KEY=STRIP(LOW2KEY,T,’x’)
                  LOW2KEY=LEFT(C2X(LOW2KEY),16)
                END
              ELSE
                DO
                  LOW2KEY=LEFT(SPACE(LOW2KEY,1,’x’),8,’x’)
                  LOW2KEY=STRIP(LOW2KEY,T,’x’)
                  LOW2KEY=LEFT(C2X(LOW2KEY),16)
                END
              END
            END
          END
        END
      END
    END
  END
END
END

END

/******* PROCESSES ALL TYPES OF HIGH2KEY VALUES *******/
DO
  IF COLTYPE='DATE' THEN
    DO
      END
    END

END

© 1998. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (940) 455 7050, fax (940) 455 2492.
HIGH2KEY=LEFT(C2X(HIGH2KEY),8)
HIGH2KEY=INSERT(HIGH2KEY,'        ')
END ELSE DO IF COLTYPE='TIMESTAMP' THEN
HIGH2KEY=LEFT(C2X(HIGH2KEY),16)
ELSE DO IF COLTYPE='TIME' THEN
DO
HIGH2KEY=LEFT(C2X(HIGH2KEY),6)
HIGH2KEY=INSERT(HIGH2KEY,'  ')
END ELSE DO IF COLTYPE='CHAR' |,
COLTYPE='VARCHAR' THEN
HIGH2KEY=LEFT(TRANSLATE(HIGH2KEY,"." ,'ØØ'x),16)
ELSE IF COLTYPE='DECIMAL' THEN
DO
HIGH2KEY=LEFT(SPACE(HIGH2KEY,1,'x'),17,'x')
HIGH2KEY=STRIP(HIGH2KEY,T,'x')
HIGH2KEY=LEFT(C2X(HIGH2KEY),17)
HIGH2KEY=STRIP(HIGH2KEY,L,'F')
ELSE IF COLTYPE='INTEGER' THEN
DO
HIGH2KEY=LEFT(SPACE(HIGH2KEY,1,'x'),8,'x')
HIGH2KEY=STRIP(HIGH2KEY,T,'x')
HIGH2KEY=LEFT(C2X(HIGH2KEY),16)
END ELSE DO
HIGH2KEY=LEFT(SPACE(HIGH2KEY,1,'x'),8,'x')
HIGH2KEY=STRIP(HIGH2KEY,T,'x')
HIGH2KEY=LEFT(C2X(HIGH2KEY),16)
END END END END

****** DISPLAYS SPECIFIC FILE RECORD *******/
DO
LIST.SCR=LEFT(TBNAME,8)'  'LEFT(NAME,18)'  'RIGHT(COLCARD,8,Ø).
' 'LOW2KEY' 'HIGH2KEY
SAY 'TABLE' COLUMN COLCARD LOW2KEY
HIGH2KEY'
SAY LIST.SCR
PARSE PULL +1Ø. +2Ø. COLCARD +8. +5 LOW2KEY +16. +1 HIGH2KEY +16.
COLTYPE.
END

/******* PREPARES CHANGED FIELDS TO REWRITE FILE RECORD ******/
IF COLCARD=''
    THEN 
    DO
        TBNAME=LEFT(TBNAME,18)
        NAME=LEFT(NAME,18)
        IF COLCARD='FFFFFFFF' &,
            DATATYPE(COLCARD)=NUM THEN
            COLCARD=D2C(COLCARD,4)
        ELSE
            IF COLCARD='FFFFFFFF' THEN
                COLCARD=COPIES(X2C('FF'),4)
            ELSE
                A=A-1
                SAY '***ERROR: COLCARD IS INVALID'
                ITERATE
            END
    END

/******* PREPARES LOW2KEY VALUES THAT WERE CHANGED ******/
DO
    LOW2KEY=STRIP(LOW2KEY,T,' ')
    IF COLTYPE = 'DATE' THEN
        DO
            LOW2KEY=STRIP(LOW2KEY,T,'x')
            LOW2KEY=LEFT(X2C(LOW2KEY),4)
        END
    ELSE
        IF COLTYPE= 'TIMESTAMP' THEN
            LOW2KEY=LEFT(X2C(LOW2KEY),8)
        ELSE
            DO
                IF COLTYPE='TIME' THEN
                    LOW2KEY=LEFT(X2C(LOW2KEY),3)
                ELSE
                    DO
                        IF COLTYPE='DECIMAL' THEN
                            LOW2KEY=INSERT('F',LOW2KEY)
                        LOW2KEY=LEFT(X2C(LOW2KEY),16)
                    END
                ELSE
                    IF COLTYPE='INTEGER' THEN


LOW2KEY=LEFT(X2C(LOW2KEY),8)
ELSE
  IF COLTYPE='SMALLINT' THEN
    LOW2KEY=LEFT(X2C(LOW2KEY),4)
  ELSE
    IF POS('.',LOW2KEY)= 0 then
      LOW2KEY=LEFT(TRANSLATE(LOW2KEY," ","x"),8)
    ELSE
      LOW2KEY=LEFT(TRANSLATE(LOW2KEY,'ØØ'x,"."),8)
  END
END
END
END

/******* PREPARES HIGH2KEY VALUES THAT WERE CHANGED ******/
DO
  HIGH2KEY=STRIP(HIGH2KEY,T,')')
  IF COLTYPE = 'DATE' THEN
    DO
      HIGH2KEY=STRIP(HIGH2KEY,T,'x')
      HIGH2KEY=LEFT(X2C(HIGH2KEY),4)
    END
  ELSE
    IF COLTYPE= 'TIMESTAMP' THEN
      HIGH2KEY=LEFT(X2C(HIGH2KEY),8)
    ELSE
      IF COLTYPE='TIME' THEN
        HIGH2KEY=LEFT(X2C(HIGH2KEY),3)
      ELSE
        DO
          IF COLTYPE='DECIMAL' THEN
            DO
              HIGH2KEY=INSERT('F',HIGH2KEY)
              HIGH2KEY=LEFT(X2C(HIGH2KEY),16)
            END
          ELSE
            IF COLTYPE='INTEGER' THEN
              HIGH2KEY=LEFT(X2C(HIGH2KEY),8)
            ELSE
              IF COLTYPE='SMALLINT' THEN
                HIGH2KEY=LEFT(X2C(HIGH2KEY),4)
              ELSE
                IF POS('.',HIGH2KEY)= 0 then
                  HIGH2KEY=LEFT(TRANSLATE(HIGH2KEY," ","x"),8)
                ELSE
                  HIGH2KEY=LEFT(TRANSLATE(HIGH2KEY,'ØØ'x,"."),8)
                END
              END
            END
      END
END
END

LIVE EXPERIENCE

In December 1997, my company implemented a totally new business, supported by a large on-line insurance application package. Because this was a new business there was hardly any data converted and the DB2 tables had very little data.

After the physical implementation of the DB2 tables, the application had very poor response times and a lot of contention between the transactions.
After some monitoring, I concluded that most of the transactions were doing sequential, list, or dynamic prefetch – not making use of the indexes available per table.

I decided to implement these programs and REXXs in the following manner (I had already executed the RUNSTATS utility on all the tablespaces, tables, and indexes before executing the BIND of the whole application):

- Execute DB2CATEX to extract the values produced by the RUNSTATS on the various tables of the catalog.
- Execute REXX DB2 DB2CATTB and change only the CARD value, table by table, according to the total number of records expected at the end of 1998, instead of the current number of records (which was very few or none at all).
- Execute REXX DB2CATIX and change the FULLKEYCARD values equal to the CARD values entered in the previous REXX, doing this index by index.
- Execute DB2CATUP to update these respective values in the SYSTABLES and SYSINDEXES catalog tables of the production environment. By setting the other files’ DD statements to dummy, the program only executes the updates for the files specified.
- Execute the BIND of the whole application with the option EXPLAIN(YES) and verify that the access paths changed from sequential and list prefetch to index accesses.

CONCLUSION

After this implementation, the application’s response times improved considerably, most of the sequential accesses changed to index accesses, and all contention between CICS transactions disappeared.

By forcing DB2 to assume these values as the real ones during the BIND process, although physically it contains other values, the DB2 Optimizer decides to use index access paths.

Iolanda Lopes  
Database Administrator  
Companhia Seguros Mundial Confiança (Portugal)  
© Xephon 1998
PLAN and PACKAGE management

INTRODUCTION
As an easy way to create the necessary statements for BIND and
REBIND PLAN/PACKAGE, I have created two batch REXX EXECs.
Starting from a DB2 catalog query, these procedures build libraries
and jobs to REBIND a full project, or BIND a new project starting
from an older one.

BIND PROCEDURE DESCRIPTION
With proper parameter customization, the tool builds libraries that
contain the following type of BIND statement:

• I – BIND PACKAGE + BIND PLAN (including PKLIST).
• II – BIND PACKAGE + BIND PLAN (including PKLIST +
  MEMBER).
• III – BIND PLAN (including MEMBER).

The following parameters describe how you can customize the REXX
EXECs:

• SUBSYS – the DB2 subsystem name (no default).
• OPER – the BIND extract type. When the value is ‘YYN’, a type
  I extraction BIND is performed. When the value is ‘YYY’, a type
  II extraction BIND is performed, and when the value is ‘NNY’,
  a type III extraction BIND is performed (no default).
• CREATOR – the creator/owner of PLAN and PACKAGE (no
  default).
• HIWORK – the high-level name for work areas (default= ‘*’;
  when the value is ‘*’, the high-level name is equal to the TSO
  user-id).
• ESOUNIT – the esoteric name for dataset allocation (default= ‘*’;
  when the value is ‘*’, the ESOUNIT is equal to WORKA).
• ACCOUNT – The account job name (no default).

Depending upon the selection made for the OPER parameter, you can have the following output datasets:

• YYY
  – hiwork.subsys.creator.PLANPACK – library with the members of BIND PLAN, including PKLIST, and, if there is mix plan, BIND PLAN including PKLIST + MEMBER.
  – hiwork.subsys.creator.PACKAGE – library with the members of BIND PACKAGE.
  – hiwork.subsys.creator.PLAN – library with the members of BIND PLAN, including MEMBER.
  – hiwork.subsys.creator.UTILITY – library which contains the utility and BIND submission jobs.

• YYN
  – hiwork.subsys.creator.PLANPACK – library with the members of BIND PLAN, including PKLIST.
  – hiwork.subsys.creator.PACKAGE – library with the members of BIND PACKAGE.
  – hiwork.subsys.creator.UTILITY – library which contains the utility and bind submission jobs.

• NNY
  – hiwork.subsys.creator.PLAN – library with the members of BIND PLAN including MEMBER.
  – hiwork.subsys.creator.UTILITY – library which contains the utility and bind submission jobs.

The procedure may be used in several ways. For example, it might be useful during the migration of an application from the development stage to the test stage, creating all you need for the BIND process. Alternatively, it may be used during a back-up to extract all you need for the BIND process.
REBIND PROCEDURE DESCRIPTION

With proper parameter customization, the tool can build jobs for REBIND PLAN and PACKAGE. The following parameters describe how you can customize the REXX EXEC:

- **SUBSYS** – the DB2 subsystem name (no default).
- **CREATOR** – the creator/owner of PLAN and PACKAGE (no default).
- **AUTOSUB** – allows you to automatically submit the jobs of Rebind PLAN/PACKAGE (default = ‘*’; when the value is ‘*’, AUTOSUB is equal to NO).
- **JOBNAME** – the job name for REBIND procedures (default = ‘*’; when the value is ‘*’, the jobname will be created automatically).
- **HIWORK** – the high-level name for work areas (default = ‘*’; when the value is ‘*’, the high-level name is equal to TSO userid).
- **ESOUNIT** – the esoteric name for dataset allocation (default = ‘*’; when the value is ‘*’, the ESOUNIT is equal to WORKA).
- **ACCOUNT** – the account job name (no default).

The procedure has been tested in an MVS Version 4.3.0 and DB2 Version 3.1 environment.

### DB2BIND REXX EXEC

```/* REXX */
trace ?o

/ * —————————————————————————————————— */
/ * — Bind Catalog Extractor for DB2 v3.1 — */
/ * —————————————————————————————————— */
/ * +— Subsys : DB2 subsystem name +— no default — */
/ * +— Oper : Bind extract type +— no default — */
/ * +— Creator : Creator/Owner +— no default — */
/ * +— Hiwork : HI-level work areas +— * = USERID — */
/ * +— Esounit: Esoteric name for allocation +— * = worka — */
/ * +— Account: Name of Job ACCOUNT +— no default — */
```

© 1998. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (940) 455 7050, fax (940) 455 2492.
```plaintext
$i  = 0
ct  = 0
ctl = 0
ctrpk = 0
ctrppk = 0
ctrppm = 0
blk =
wrk =
swsyspr = off
yesppk = off
jna = userid()
user = userid()
notif = userid()
arg parmin
parm   = translate(parmin,' ',',')
nparm  = words(parm)
subsys = word(parm,1)
oper   = word(parm,2)
creator = word(parm,3)
hiwork = word(parm,4)
esounit = word(parm,5)
account = word(parm,6)
if nparm < 6 then do
  say '';
  say '';
  say '>>>>>>>>';
  say '>>>>>>>> Parameter string is incomplete !!!!!!'
  say '>>>>>>>>' 'parmin
  say '>>>>>>>>';
  say '';
  say '';
exit
end
loper = length(oper)
if loper ¬= 3 then do
  say '';
  say '';
  say '>>>>>>>>';
  say '>>>>>>>> The variable OPER is wrong !!!!!!'
  say '>>>>>>>> it must be 3 bytes long.';
  say '>>>>>>>>'
  say '';
  say '';
exit
```

end
if oper ≠ 'YYN' &,
oper ≠ 'YYY' &,
oper ≠ 'NNY' then do
say '     
say '     
say '>>>>>>>>
say '>>>>>>>> The variable OPER is wrong !!!!!        
say '>>>>>>>> Specify :        
say '>>>>>>>>
say '>>>>>>>>  1) YYN Package + Plan with ( Pklist )        
say '>>>>>>>>  2) YYY Package + Plan with ( Pklist/Member )    
say '>>>>>>>>  3) NNY Plan with ( Member )        
say '>>>>>>>>
say '     
say '     
exit
end
if hiwork = '*' then
hiwork = userid()
if esounit = '*' then
esounit = worka
/*——— File ipoupdte allocation ———*/
/*—————————————————————————————————————————————————————————*/
outdsupd= hiwork.'subsys'. 'creator'.UTILITY'
xx=outtrap(trp01.)
address tso "delete "outdsupd""
"alloc da("outdsupd") dir(40) space(15,45) dsorg(ps)"
, "recfm(f,b) lrecl(80) blksize(27920) tracks ",
"unit("esounit") new catalog f(fiupd) "
xx=outtrap(off)
if rc > 0 then do
do #a = 1 to trp01.0
   say trp01.#a
end
say '     
say '     
say '>>>>>>>>
say '>>>>>>>> "outdsupd" Allocation OK'
say '>>>>>>>> RC='rc'. Verify.        
say '>>>>>>>>
say '     
say '     
exit
end
else
say '>>>>>>>> 'outdsupd ' Allocation OK ' /*—————————————————————————————————————————————————————————*/
/*— File systsinp allocation ———*/
/*————————————————————————————————————————————*/
outds1= hiwork'.subsys'.creator'.SYSTsinp'
xx= outtrap(trpØ2.)
   address tso "delete "outds1""
   "alloc da("outds1") dir(Ø) space(5,1) dsorg(ps)" ,
   "recfm(f,b) lrecl(Ø) blksiz(27920) tracks ",
   "unit("esounit") release new catalog f(systsinp) "
xx=outtrap(off)
if rc > Ø then do
   do #a = 1 to trpØ2.Ø
      say trpØ2.#a
   end
   say '        '
   say '        '
   say '>>>>>>>>'
   say '>>>>>>>> "outds1" Allocation OK'
   say '>>>>>>> RC='rc'. Verify. '
   say '>>>>>>>'
   say '        '
   say '        '
   exit
end
else
   say '>>>>>>> "outds1" Allocation OK '
   sk.1='DSN SYSTEM('subsys')                                           '
   sk.2='RUN PROGRAM(DSNTEP2) PLAN(DSNTEP31) LIB('DSN31Ø.RUNLIB.LOAD')'
   sk.3='END                                                            '
   sk.Ø=3;
   "execio * diskw systsinp (stem sk. finis"
call Pulisci
   /*————————————————————————————————————————————*/
   /*—     Number of select to do             ———*/
   /*————————————————————————————————————————————*/
   wrk = translate(oper,'Ø','N')
   wrk = translate(wrk,'1','Y')
   do #y = 1 to 3
      ct = substr(wrk,#y,1)
      if ct = 1 then
         ct1 = ct1 + 1
      end
   select
      when ct1 = 1 then do
         call Allppm
         /*————————————————————————————————————————————*/
         /*—     Bind Plan with member               ———*/
         /*————————————————————————————————————————————*/
         tipsel = BPMEMB
         call Allsysp
         call Allsysi
         jobw = sysin
"alloc da('"outds0"') f("jobw") shr reuse"

   sk.1=' SELECT * FROM VBPMEMB ORDER BY 1,2 ;
   sk.0=1
   call Writeout
   #i = 1
   call Rundb2
   macnr = @mdb2023
   call Exmacro
   call Testout
   if build = yes then do
      call Wrbpmemb
      say '   ';
      say '>>>>>>>>' ;
      say '>>>>>>>> Total Bind Plan "DBRM" no' ctrppm  ;
      say '>>>>>>>>' ;
      say '   ';
address tso "delete "outds0"
address tso "delete "outds2"
   end
else do
   say '   ';
   say '>>>>>>>>' ;
   say '>>>>>>>> The select BPMEMB has 0 rows !!! ' ;
   say '>>>>>>>> The 'outdssppm' will be empty. ';
   say '>>>>>>>>' ;
say '   ';
address tso "delete "outds0"
address tso "delete "outds2"
   end
end
when ct1 = 2 then do
   call Alppk
   call Alpk
   do #i = 1 to 2
      if #i = 1 then do
         /*—————————————————————————————————*/
         /*—      Bind Plan with Package            ———*/
         /*—————————————————————————————————*/
         tipsel = BPPACK
         call Allsysp
         call Allsysi
         jobw = sysin
         "alloc da('"outds0"') f("jobw") shr reuse"
         sk.1=' SELECT * FROM VBPPACK ORDER BY 2,1 ;   '
      end
end
sk.0=1
Call Writeout
Call Rundb2
macnr = @mdb2018
call Exmacro
call Testout
if build = yes then do
call Wrbppack
say '
 say '>>>>>>>>>
 say '>>>>>>>>> +——————————————————————————————————+
 say '>>>>>>>>> Total Bind Plan "Pklist" no' ctrppk
 say '>>>>>>>>> +——————————————————————————————————+
 say '>>>>>>>>>
say'
address tso "delete '"outdsØ"'"
address tso "delete '"outds2"'"
say'
say'
say'
end
else do
say'
 say '>>>>>>>>>
say '>>>>>>>>> The select BPPACK has Ø rows !!! '
say '>>>>>>>>> The 'outdsppk' will be empty,    '
say '>>>>>>>>>
say'
address tso "delete '"outdsØ"'"
address tso "delete '"outds2"'"
say'
end
if #i = 2 then do
  /*— Bind Package ———*/
  tipsel = BPACK
call Allsysp
call Allsysi
jobw = sysin
"alloc da("outdsØ") f("jobw") shr reuse"
   sk.1= ' SELECT * FROM VBPACK ORDER BY 2,1 ;   '
sk.0=1
call Writeout
call Rundb2
macnr = @mdb2016
call Exmacro
call Testout
if build = yes then do
call Wrbpack
if #pk > 0 then do
  say '        '
  say '>>>>>>>>>
  say '>>>>>>>>> Warning there are no '#pk' packages
      with member name PK???????'
  say '>>>>>>>>> because the name of Package is
      not unique !!!!'
  say '>>>>>>>>>
end
say '        '
say '>>>>>>>>>
say '>>>>>>>>> +——————————————————————————————+
say '>>>>>>>>>    Total Bind Package no' ctrpk
say '>>>>>>>>> +——————————————————————————————+
say '        '
address tso "delete '"outdsØ"'''
address tso "delete '"outds2"'''
say '        '
end
else do
  say '        '
  say '>>>>>>>>>
  say '>>>>>>>>> The select BPACK has Ø rows !!! '
  say '>>>>>>>>> The 'outdspk' will be empty.    '
  say '>>>>>>>>>
  say '        '
  address tso "delete '"outdsØ"'''
  address tso "delete '"outds2"'''
say '        '
end
end
when ct1 = 3 then do
  call Allppk
  call Allpk
  call Allppm
  do #i = 1 to 3
    if #i = 1 then do
  /*— Bind Plan with Package ———*/
  / *—*/
  tipsel = BPPACK
  call Allsysp
  call Allsysi
  jobw = sysin
  "alloc da(’”outdsØ”’) f(”jobw”) shr reuse"
  sk.l=’ SELECT * FROM VBPPACK ORDER BY 2,1 ;    ’
sk.0=1
call Writeout
call Rundb2
macnr = @mdb2018
call Exmacro
call Testout
if build = yes then do
call Wrbppack
say '   
say '<<<<<<>'
say '<<<<>>>>> +———————————————————————————————————+'
say '<<<<>>>>> Total Bind Plan "Pklist" no' ctrppk
say '<<<<>>>>> +———————————————————————————————————+'
say '<<<<>>>>>

say '   
address tso "delete '"outdsØ"''
address tso "delete '"outds2"''
say '   
end
else do
say '   
say '<<<<>>>>>

say '<<<<>>>>> The select BPPACK has Ø rows !!! '
say '<<<<>>>>> The 'outdssppk' will be empty.    '
say '<<<<>>>>>

say '   
address tso "delete '"outdsØ"''
address tso "delete '"outds2"''
say '   
end
if #i = 2 then do
/*————————————————————————————————————————————*/
/*—      Bind Package                      ———*/
/*————————————————————————————————————————————*/
tipsel = BPACK
call Allsysp
call Allsysi
jobw = sys1n
"alloc da(""outdsØ"") f("jobw") shr reuse"
    sk.l=' SELECT * FROM VBPACK ORDER BY 2,1 ;    '
sk.Ø=1
call Writeout
call Rundb2
macnr = @mdb2016
call Exmacro
call Testout
if build = yes then do
call Wrbppack
if #pk > Ø then do
say '>>>>>>>>
say '>>>>>>>> Warning there are no '#pk' packages
    with member name PK??????'
say '>>>>>>>> because the name of Package is
    not unique !!!!!!'
say '>>>>>>>>
end
say '>>>>>>>>
say '>>>>>>>> +——————————————————————————————+
say '>>>>>>>>    Total Bind Package no' ctrpk
say '>>>>>>>> +——————————————————————————————+
say '>>>>>>>>
address tso "delete "outdsØ""
address tso "delete "outds2"
say '        
end
else do
say '        
say '>>>>>>>>
say '>>>>>>>> The select BPACK has Ø rows !!!'
say '>>>>>>>> The 'outdspk' will be empty.    
say '>>>>>>>>
say '        
address tso "delete "outdsØ"
address tso "delete "outds2"
say '        
end
if #i = 3 then do

    /*————————————————————————————————————————————*/
    /*—      Bind Plan with member (Mix Plan)  ———*/
    /*————————————————————————————————————————————*/

    tipsel = BPMEMB
call Allsysp
call Allsysi
jobw = sysin
"alloc da("outdsØ") f("jobw") shr reuse"
    sk.1=' SELECT * FROM VBPMEMB ORDER BY 1,2 ;   '
    sk.Ø=1
call Writeout
call Rundb2
macnr = @mdb2Ø23
call Exmacro
call Testout
if build = yes then do
call Wrbpmemb
say '        '
say '>>>>>>>>'  
say '>>>>>>>>  Total Bind Plan "DBRM" no' ctrppm 
say '>>>>>>>>'  
say '>>>>>>>>'  

describe tso "delete '"outds0"'"  
describe tso "delete '"outds2"'"  
say '  

/*————————————————————————————————————————————*/  
/*—      File out compare allocation       ———*/  
/*————————————————————————————————————————————*/  
  if yesppk = on then do 
    call Alldelt  
/*————————————————————————————————————————————*/  
/*—      List member .PLANPACK             ———*/  
/*————————————————————————————————————————————*/  
    xx=outtrap(trpdl.)  
    describe tso "listds '"outdsppk"' members "  
    xx=outtrap(off)  
    jobw = newdd  
    "alloc da("outdsupd"($PACK)) f("jobw") shr reuse"  
    call wrcomp  
/*————————————————————————————————————————————*/  
/*—      List member .PLAN                 ———*/  
/*————————————————————————————————————————————*/  
    xx=outtrap(trpdl.)  
    describe tso "listds '"outdsppm"' members "  
    xx=outtrap(off)  
    jobw = olddd  
    "alloc da("outdsupd"($PLAN)) f("jobw") shr reuse"  
    call wrcomp  
/*————————————————————————————————————————————*/  
/*—      Compare member                    ———*/  
/*————————————————————————————————————————————*/  
  "ispexec select pgm(isrsupc) parm(CHNGL,LINECMP)"  
  if rc > 4 then do  
    say '  
    say '  
    say '>>>>>>>>'  
    say '>>>>>>>>  Compare dataset error '  
    say '>>>>>>>>  RC = 'rc'. Verify. '  
    say '>>>>>>>>'  
    say '  
    say '  
    call free  
    exit  
end  
/*————————————————————————————————————————————*/  
/*—    Macro to process out compare        ———*/
/*————————————————————————————————————————————*/
xx=OUTTRAP(trpØ3.)
"ispexec edit dataset('"outdsdlt"') macro(@MDB2Ø27)"
xx=OUTTRAP(OFF)
if rc > Ø then do
  do #a = 1 to trpØ3.Ø
    say trpØ3.#a
  end
  exit
end
/*————————————————————————————————————————————*/
/*—      Extract member name of mix plans ———*/
/*————————————————————————————————————————————*/
xx=outtrap(trpØ4.)
"execio * diskr outdd (stem outdd. finis"
"free fi(outdd)"
xx=outtrap(off)
if rc > Ø then do
  do #a = 1 to trpØ4.Ø
    say trpØ4.#a
  end
  say '        ' say '        ' say '        ' say '        ' say '        ' say '        ' say '        ' say '        ' say '        ' exit
end
if outdd.Ø = Ø then do
  say '        ' say '        ' say '        ' say '        ' say '        ' say '        ' say '        ' say '        ' exit
end
/*————————————————————————————————————————————*/
/*—      Member name library list           ——*/
/*————————————————————————————————————————————*/
#o = Ø
#k = Ø
sw = off
do while mem_com ¬= Z9999999
  #k = #k + 1
mem_com = word(outdd.#k,1)
tst_com = substr(outdd.#k,2,4)
if sw = on then do
    if tst_com = blk then do
        #o = #o + 1
        arr.#o = mem_com
    end
end
if mem_com = $$$COIBM then
    sw = on
end
/*————————————————————————————————————————————*/
/*—      If there are no mix plans         ———*/
/*————————————————————————————————————————————*/
select
    when #o = Ø | #o = 1 then do
        say '        '
say '        '
say '>>>>>>>>'
say '>>>>>>>>  There are no mix plan (pklist + member). '
say '>>>>>>>>'
say '        '
address tso "delete ""outdsupd"($PLAN)"
address tso "delete ""outdsupd"($PACK)"
address tso "delete ""outdsspkm"($PACK)"
address tso "delete ""outdsspkm"($PACK)"
end
when #o > 1 then do
    say '>>>>>>>>'
say '>>>>>>>>  Building mix plans (Member + pklist).'
say '>>>>>>>>'
say '        '
do #z = 1 to #o - 1
    xx=OUTTRAP(trpØ5.)
    "ispexec edit dataset(""outdsspkm"("arr.#z")")"
    macro(@mdb2Ø21)"
    xx=OUTTRAP(OFF)
    if rc > Ø then do
        do #a = 1 to trpØ5.Ø
            say trpØ5.#a
        end
        exit
    end
    "ispexec lminit dataid(""inpl")"
    dataset(""outdsspkm") enq(shr)"
    "ispexec lminit dataid(""out1")"
    dataset(""outdsspkm") enq(shr)"
    lwrk = length(arr.#z)
    movmemb = # || substr(arr.#z,2,lwrk)
"ispexec lmmove fromid("inpl") frommem("arr.#z")
todataid("out1") tomem("movmemb")"
x=x=OUTTRAP(trpØ6.)
address ispexec 'vput (movmemb) profile'
"ispexec edit dataset('"outdsppk"("arr.#z")')
macro(@mdb2Ø24)"
x=x=OUTTRAP(OFF)
if rc > Ø then do
do #a = 1 to trpØ6.Ø
  say trpØ6.#a
end
exit
end

say '>>>>>>>>>

say '>>>>>>>>> There are '#o - 1' mix plans.'
say '>>>>>>>>>

say '        

/*————————————————————————————————————————————*/
/*—      List member .PLAN                 ———*/
/*————————————————————————————————————————————*/

address tso "delete '"outdsupd"($PLAN)"'
address tso "delete '"outdsupd"($PACK)"
address tso "delete '"outdsppk"(Z9999999)"
address tso "delete '"outdsppm"(Z9999999)"
x=x=outtrap(trpdl.)
  address tso "listds '"outdsppm"' members "
x=x=outtrap(off)
jobw = olddd
"alloc da '"outdsupd"($PLAN)" f("jobw") shr reuse"
#x  = 1
#x1 = Ø
#r  = Ø
call hdrbnd
sw= off
do #h = 1 to trpdl.Ø
  if sw = on then do
    wrk = word(trpdl.#h,1)
    #r = #r + 1
    if #r > 1 then do
      if swsyspr = on then do sb.#x='//SYSTSIN
        DD  DSN='outdsppm'('wrk'),DISP=SHR  '
        #x = #x + 1
        swsyspr = off
      end
      else do sb.#x='// DD  DSN='outdsppm
        '('wrk'),DISP=SHR   '
        #x = #x + 1
    end
  end
end
if #x1 > 8Ø then dosb.#x='  
  /* ————————————————————————— *  
  #x = #x + 1  
call hdrbnd  
  #x1 = Ø  
end  
  #x1 = #x1 + 1  
end  
end  
if trpdl.#h = '—MEMBERS—' then  
  sw = on  
end sb.#x='  
  /* ————————————————————————— *  
  sb.Ø = #x  
jobw = fiupd  
"alloc da('"outdsupd"(bindppm)) f("jobw") shr reuse"  
call Wrindx  
if #r = 1 then do  
  address tso "delete '"outdsppm"'"  
  address tso "delete '"outdsupd"(bindppm)'"  
  address tso "delete '"outdsupd"(ipoupppm)'"  
end  
otherwise  
  say '  
  say '  
  say '>>>>>>>>'  
say '>>>>>>>> Unpredictable error. Stop elaboration !!!! '  
  say '>>>>>>>>'  
  say '  
  say '  
call free  
exit  
end  
address tso "delete '"outdsdlt"'"  
end  
else do  
  say '  
  say '>>>>>>>>'  
say '>>>>>>>> The select BPMEMB has Ø rows !!! '  
say '>>>>>>>> The 'outdsppm' will be empty. '  
say '>>>>>>>>'  
say '  
  address tso "delete '"outdsØ"'"  
  address tso "delete '"outds2"'"  
  say '  
end  
end  
end
otherwise
    say '        '    
say '        '    
say '>>>>>>>>'    
say '>>>>>>>> Unpredictable OPER value. Stop elaboration !!!!! '    
say '        '    
say '        '
end
call Free
exit

/*————————————————————————————————————————————*/
/*—     File Plan allocation (whit packlist) —*/
/*————————————————————————————————————————————*/

Allppk:
outdsppk= hiwork.'subsys'.creator'.PLANPACK'
x=trap(trp07.)
    address tso "delete ''outdsppk''"
    "alloc da('''outdsppk''') dir(900) space(150,450) dsorg(ps)"
    "recfm(f,b) lrecl(80) blksize(27920) tracks",
    "unit("esounit") release new catalog f(fippk)"
x=trap(off)
if rc > 0 then do
    do #a = 1 to trp07.0
        say trp07.#a
    end
    say '        '    
say '        '    
say '>>>>>>>>'    
say '>>>>>>>> ''outdsppk'' Allocation OK'    
say '>>>>>>>> RC='rc'. Verify.    
say '>>>>>>>>'    
say '        '    
exit
end
else
    say '>>>>>>>> 'outdsppk ' Allocation OK '    
return

/*—————————-——————————————————————————————————*/
/*—       File Package allocation          ———*/
/*————————————————————————————————————————————*/

Allpk:
outdspk= hiwork.'subsys'.creator'.PACKAGE'
x=trap(trp08.)
    address tso "delete ''outdspk''"
    "alloc da('''outdspk''') dir(2000) space(150,900) dsorg(ps)"
    "recfm(f,b) lrecl(80) blksize(27920) tracks",
    "unit("esounit") release new catalog f(fipk)"
x=trap(off)
if rc > 0 then do
  do #a = 1 to trp08.0
    say trp08.#a
  end
  say '   
  say '   
  say '>>>>>>>>>
  say '>>>>>>>>> "outdspk" Allocation OK'
  say '>>>>>>>>> RC='rc'. Verify. 
  say '>>>>>>>>>
  say '   
  say '   
  exit
end
else
  say '>>>>>>>>> 'outdspk' Allocation OK' 
  return
/*————————————————————————————————————————————*/
/*—  File Plan allocation (with member) ——*/
/*————————————————————————————————————————————*/
Allppm:
  outdsppm= hiwork.'subsys'.'creator'.PLAN'
  xx=outtrap(trp09.)
    address tso "delete "outdsppm"
    "alloc da("outdsppm") dir(900) space(150,450) dsorg(ps)", 
    "reclm(80) lrecl(80) blksz(27920) tracks ", 
    "unit('esounit') release new catalog f(fippm) "
  xx=outtrap(off)
if rc > 0 then do
  do #a = 1 to trp09.0
    say trp09.#a
  end
  say '   
  say '   
  say '>>>>>>>>>
  say '>>>>>>>>> "outdsppm" Allocation OK'
  say '>>>>>>>>> RC='rc'. Verify. 
  say '>>>>>>>>>
  say '   
  say '   
  exit
end
else
  say '>>>>>>>>> 'outdsppm' Allocation OK' 
  return
/*————————————————————————————————————————————*/
/*—      File sysin allocation ———*/
/*————————————————————————————————————————————*/
Allsysi:
  outdsØ= hiwork.'subsys'.'creator'.'tipsel'.SYSIN'
xx=outtrap(trp10.)
  address tso "delete "outds0"
  "alloc da("outds0") dir(Ø) space(5,1) dsorg(ps)",
  "recfm(f,b) lrecl(80) blksiz(27920) tracks ",
  "unit("esounit") release new catalog f(sysin)"
xxt=delete 'outds0'
  "alloc da('outds0') dir(Ø) space(5,1) dsorg(ps)",
  "recfm(f,b) lrecl(80) blksiz(27920) tracks ",
  "unit("esounit") release new catalog f(sysin)"
xxt=delete 'outds0'
if rc > Ø then do
  do #a = 1 to trp10.Ø
    say trp10.#a
  end
  say '        '
  say '        '
  say '>>>>>>>>'
  say '>>>>>>>> "outds0" Allocation OK'
  say '>>>>>>>> RC='rc'. Verify.  ,
  say '        '
  say '        '
  exit
else do
  say '>>>>>>>> "outds0" Allocation OK '
  say '        '
end
return
/*@—    File sysprint allocation ———*/
/*@—    File sysprint allocation ———*/
Allsysp:
  outds2= hiwork.'subsys.'creator.'tipsel'.SYSPRINT'
xxt=delete 'outds2'
  "alloc da("outds2") dir(Ø) space(350,350) dsorg(ps)",
  "recfm(f,b,a) lrecl(121) blksiz(1210) tracks ",
  "unit("esounit") release new catalog f(sysprint)"
xxt=delete 'outds2'
if rc > Ø then do
  do #a = 1 to trp11.Ø
    say trp11.#a
  end
  say '        '
  say '        '
  say '>>>>>>>>'
  say '>>>>>>>> "outds2" Allocation OK'
  say '        RC='rc'. Verify.  ,
  say '        '
  say '        '
  exit
else
Rundb2:

wrk = center(tipsel,8)
say ' >>>>>>>>>>>  'outds2 '  Allocation OK  'return

/*————————————————————————————————————————————*/
/*—      Run DB2                           ———*/
/*————————————————————————————————————————————*/
Rundb2:

wrk = center(tipsel,8)
say ' >>>>>>>>>>>  'outds2 '  Allocation OK  'return

/*————————————————————————————————————————————*/
/*—      Print output DB2 query            ———*/
/*————————————————————————————————————————————*/
address tso "pr dataset('"outds2") class(R) writer(nØz2) fcb(6) "return

/*————————————————————————————————————————————*/
/*—    Macro to process output of sysprint  ——*/
/*————————————————————————————————————————————*/
Exmacro:

say ' >>>>>>>> >  'outds1 '  Allocation OK  'return

/*————————————————————————————————————————————*/
/*—      Run DB2                           ———*/
/*————————————————————————————————————————————*/
xx=OUTTRAP(trp13.)
"ispexec edit dataset('"outds2"') macro('"macnr"')"
xx=OUTTRAP(OFF)
if rc > Ø then do
   do #a = 1 to trp13.Ø
      say trp13.#a
   end
   exit
end
return

/*————————————————————————————————————————————*/
/*—      Test output DB2 query             ———*/
/*————————————————————————————————————————————*/
Testout:
build = yes
xx=outtrap(trp14.)
"execio * diskr sysprint (stem sysprint. finis"
"free fi(sysprint)"
xx=outtrap(off)
if rc > Ø then do
   do #a = 1 to trp14.Ø
      say trp14.#a
   end
   say '   ' '   
   say '>>>>>>>'
   say '>>>>>>>' Error reading file "'outds2'"
   say '>>>>>>>' RC='rc'. Verify. 
   say '>>>>>>>'
   say '   ' '   
   exit
end
if sysprint.Ø = Ø then do
   say '   ' '   
   say '>>>>>>>'
   say '>>>>>>>' The file "'outds2'" is empty.  ' 
   say '>>>>>>>' Probably error accessing DB2 !!!!!!
   say '>>>>>>>'
   say '   ' '   
   exit
end
if substr(sysprint.1,33,8) = ' Ø ROW(S)' then
   build = no
return

/*————————————————————————————————————————————*/
/*—      Build members Bind Package        ———*/
/*————————————————————————————————————————————*/
Wrtpack:

say '>>>>>>>>> Building library 'outdspk
jobw = fipk
"alloc da('"outdspk"($$coibm)') f("jobw") shr reuse"

sk.1='/////////////////////////////////////////////////////'
sk.2='****** Do not erase this member !!!! Thanks. ****** '
sk.3='/////////////////////////////////////////////////////

sk.Ø = 3
call Writeout
#pk = Ø
#x  = 1
#x1 = Ø

call Hdrbnd

DO #b = 1 to sysprint.Ø
    pak_pk = word(sysprint.#b,1)
    col_pk = word(sysprint.#b,2)
    own_pk = word(sysprint.#b,3)
    qua_pk = word(sysprint.#b,4)
    val_pk = word(sysprint.#b,5)
    exp_pk = word(sysprint.#b,6)
    pds_pk = word(sysprint.#b,7)
    iso_pk = word(sysprint.#b,8)
    rel_pk = word(sysprint.#b,9)
    if substr(sysprint.#b,122,1) = blk then
        iso_pk = S
    if substr(sysprint.#b,127,1) = blk then
        rel_pk = C
    /*— VALIDATE "Bind/Run" ———*/
    if val_pk = B then
        valid = BIND
    else
        valid = RUN
    /*— ISOLATION "Rep.Read/Cursor Stability ——*/
    if iso_pk = R then
        isol = RR
    else
        isol = CS

Editor's note: this article will be continued next month.

Giuseppe Rendano
DB2 Systems Programmer (Italy) © Xephon 1998
IBM has announced Version 1.5 of DFSMS/MVS, promising less overhead and redundancies than in earlier versions of HFS. In this new release, OAM will exploit sysplex architecture and DB2 data sharing, and will remove the 100 storage group limitation in a configuration. Users can now specify their own high-level qualifier for the object groups and their associated DB2 tables.

OAM will also exploit sysplex architecture to enhance availability and allow instances of OAM which belong to an XCF group and are connected to DB2 subsystems, using DB2 data sharing to have full access to OAM objects.

For further information contact your local IBM representative.

* * *

ETI has announced Data System Library (DSL) for COBOL/DB2 Release 1.2.1. Developed jointly with IBM, DSL provides support for IBM DB2 Universal Database (including the Extended Enterprise Edition with parallel loading), DB2 for OS/400, DB2 for MVS and OS/390, with support for parallel sysplex. It offers the ability to load data into any DB2-based application or data warehouse from any number of different sources. DSL can be used in conjunction with The MetaTransport Utility for Release 1.0.0 which integrates the ETIïExtract Tool Suite with IBM’s Visual Warehouse datamart and data warehouse software.

For further information contact:
ETI, 66 Bovet Road, Suite 320, San Mateo, CA 94402, USA.
Tel: (650) 345 9100.

* * *

IBM is to include Platinum’s ERwin 3.5 DB2 data modelling tool, for the design, generation, and maintenance of database applications, as part of its VisualAge family. The companies plan to sell an enhanced version of ERwin, by the end of the year, that will integrate additional functions and features of VisualAge DataAtlas.

Further collaboration is to focus on an open, industry standard XML interface, which will integrate ERwin into VisualAge through TeamConnection, IBM’s workgroup software. This will give development teams the ability to model databases with ERwin and store and maintain them in the VisualAge TeamConnection repository where other application code resides.

For further information contact your local IBM representative.

* * *

---

xephon